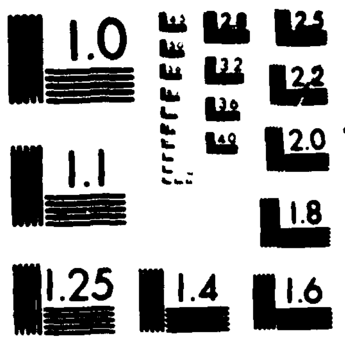
AD-A183 552 TEN PROBLEMS IN ARTIFICIAL INTELLIGENCE(U) YALE UNIV NEW HAVEN CT DEPT OF COMPUTER SCIENCE R C SCHANK ET AL. JAN 87 YALEU/CSD/RR-514 1/1

UNCLASSIFIED N00014-85-K-0108 F/G 12/9 NL

END
9-87
DTIC



MICROCOPY RESOLUTION TEST CHART

NATIONAL BUREAU OF STANDARDS 1963-A

Ten Problems in Artificial Intelligence

Roger C. Schank   Christopher C. Owens

YALEU/CSD/RR #514

January 1987

YALE UNIVERSITY
DEPARTMENT OF COMPUTER SCIENCE

Yale University
Department of Computer Science

Ten Problems in Artificial Intelligence

Roger C. Schank Christopher C. Owens

YALEU/DCS/TR-514

February 1987

This work is supported in part by the Air Force Office of Scientific Research under grant AFOSR-85-0343, and in part by the Defense Advanced Research Projects Agency under Office of Naval Research grant N00014-85-K-0108.

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER #514	2. GOVT ACCESSION NO. AD-A18[illegible]	3. RECIPIENT'S CATALOG NUMBER
4. TITLE *(and Subtitle)* Ten Problems in Artificial Intelligence		5. TYPE OF REPORT & PERIOD COVERED Research Report
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Roger C. Schank and Christopher C. Owens		8. CONTRACT OR GRANT NUMBER(s) N00014-85-K-0108
9. PERFORMING ORGANIZATION NAME AND ADDRESS Yale University - Computer Science Department 10 Hillhouse Avenue New Haven, CT 06520		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Advanced Research Projects Agency 1400 Wilson Boulevard Arlington, VA 22209		12. REPORT DATE January 1987
		13. NUMBER OF PAGES 28
14. MONITORING AGENCY NAME & ADDRESS *(if different from Controlling Office)* Office of Naval Research Information Systems Program Arlington, VA 22217		15. SECURITY CLASS. *(of this report)* unclassified
		15a. DECLASSIFICATION DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT *(of this Report)*

Approved for public release; distribution unlimited

17. DISTRIBUTION STATEMENT *(of the abstract entered in Block 20, if different from Report)*

18. SUPPLEMENTARY NOTES

19. KEY WORDS *(Continue on reverse side if necessary and identify by block number)*

Artificial Intelligence
Research methodology
Philosophical Foundation
Fundamental Issues

20. ABSTRACT *(Continue on reverse side if necessary and identify by block number)*

Researchers in Artificial Intelligence have had a difficult time defining the field's goals and assessing its progress. Some have focused on the task of modelling the human brain, others have focused on developing smart machines independent of the constraints of psychological or neurological realism. Over the years the notion of what is an AI task has changed, as problems once thought to be easy have turned out to be hard, and vice versa.

DD FORM 1 JAN 73 1473

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

This paper discusses some problems that are currently of interest to th field, and places them in the context of a more enduring question: "What is intelligence?" It attempts to enumerate a few essential aspects of intelligence that every human, animal or intelligent machine must, to soem degree, exhibit.

OFFICIAL DISTRIBUTION LIST

Defense Documentation Center Cameron Station Alexandria, Virginia 22314	12 copies
Office of Naval Research Information Systems Program Code 437 Arlington, Virginia 22217	2 copies
Dr. Judith Daly Advanced Research Projects Agency Cybernetics Technology Office 1400 Wilson Boulevard Arlington, Virginia 22209	3 copies
Office of Naval Research Branch Office - Boston 495 Summer Street Boston, Massachusetts 02210	1 copy
Office of Naval Research Branch Office - Chicago 536 South Clark Street Chicago, Illinois 60615	1 copy
Office of Naval Research Branch Office - Pasadena 1030 East Green Street Pasadena, California 91106	1 copy
Mr. Steven Wong New York Area Office 715 Broadway - 5th Floor New York, New York 10003	1 copy
Naval Research Laboratory Technical Information Division Code 2627 Washington, D.C. 20375	6 copies
Dr. A.L. Slafkosky Commandant of the Marine Corps Code RD-1 Washington, D.C. 20380	1 copy
Office of Naval Research Code 455 Arlington, Virginia 22217	1 copy

Office of Naval Research 1 copy
Code 458
Arlington, Virginia 22217

Naval Electronics Laboratory Center 1 copy
Advanced Software Technology Division
Code 5200
San Diego, California 92152

Mr. E.H. Gleissner 1 copy
Naval Ship Research and Development
Computation and Mathematics Department
Bethesda, Maryland 20084

Captain Grace M. Hopper, USNR 1 copy
Naval Data Automation Command, Code 00H
Washington Navy Yard
Washington, D.C. 20374

Dr. Robert Engelmore 2 copies
Advanced Research Project Agency
Information Processing Techniques
1400 Wilson Boulevard
Arlington, Virginia 22209

Professor Omar Wing 1 copy
Columbia University in the City of New York
Department of Electrical Engineering and
Computer Science
New York, New York 10027

Office of Naval Research 1 copy
Assistant Chief for Technology
Code 200
Arlington, Virginia 22217

Computer Systems Management, Inc. 5 copies
1300 Wilson Boulevard, Suite 102
Arlington, Virginia 22209

Ms. Robin Dillard 1 copy
Naval Ocean Systems Center
C2 Information Processing Branch (Code 8242)
271 Catalina Boulevard
San Diego, California 92152

Dr. William Woods 1 copy
BBN
50 Moulton Street
Cambridge, MA 02138

Professor Van Dam Dept. of Computer Science Brown University Providence, RI 02912	1 copy
Professor Eugene Charniak Dept. of Computer Science Brown University Providence, RI 02912	1 copy
Professor Robert Wilensky Univ. of California Elec. Engr. and Computer Science Berkeley, CA 94707	1 copy
Professor Allen Newell Dept. of Computer Science Carnegie-Mellon University Schenley Park Pittsburgh, PA 15213	1 copy
Professor David Waltz Univ. of Ill at Urbana-Champaign Coordinated Science Lab Urbana, IL 61801	1 copy
Professor Patrick Winston MIT 545 Technology Square Cambridge, MA 02139	1 copy
Professor Marvin Minsky MIT 545 Technology Square Cambridge, MA 02139	1 copy
Professor Negroponte MIT 545 Technology Square Cambridge, MA 02139	1 copy
Professor Jerome Feldman Univ. of Rochester Dept. of Computer Science Rochester, NY 14627	1 copy
Dr. Nils Nilsson Stanford Research Institute Menlo Park, CA 94025	1 copy

Dr. Alan Meyrowitz 1 copy
Office of Naval Research
Code 437
800 N. Quincy Street
Arlington, VA 22217

LCOL Robert Simpson 1 copy
IPTO-DARPA
1400 Wilson Blvd
Arlington, VA 22209

Dr. Edward Shortliffe 1 copy
Stanford University
MYCIN Project TC-117
Stanford Univ. Medical Center
Stanford, CA 94305

Dr. Douglas Lenat 1 copy
Stanford University
Computer Science Department
Stanford, CA 94305

Dr. M.C. Harrison 1 copy
Courant Institute Mathematical Science
New York University
New York, NY 10012

Dr. Morgan 1 copy
University of Pennsylvania
Dept. of Computer Science & Info. Sci.
Philadelphia, PA 19104

Mr. Fred M. Griffee 1 copy
Technical Advisor C3 Division
Marine Corps Development
  and Education Command
Quantico, VA 22134

Dr. Vince Sigilitto 1 copy
Program Manager
AFOSR/NM
Bolling Airforce Base
Building 410
Washington, DC 20332

Abstract

Researchers in Artificial Intelligence have had a difficult time defining the field's goals and assessing its progress. Some have focused on the task of modelling the human brain, others have focused on developing smart machines independent of the constraints of psychological or neurological realism. Over the years the notion of what is an AI task has changed, as problems once thought to be easy have turned out to be hard, and vice versa. This paper discusses some problems that are currently of interest to the field and places them in the context of a more enduring question: "What is intelligence?" It attempts to enumerate a few essential aspects of intelligence that every human, animal or intelligent machine must, to some degree, exhibit.

Introduction

Once, many people thought that the ability to play chess epitomized the kind of intelligence a machine should have. Chess playing, it seemed, took genuine intelligence that couldn't be simulated via any kind of trickery. If a machine could be made to play chess it would necessarily embody something interesting in the way of intelligence. Now, however, machines are available for a few hundred dollars that play a good game of chess, and the best machines play the game at a world-class level. Yet nobody would call these machines truly intelligent in any general sense. As an indicator of general intelligence, chess-playing skill has been demystified.

Another task that, like chess playing, seems to be a mark of general intelligence, is the ability to translate text from one language to another. A machine solution to this problem has been elusively "just around the corner" since the earliest days of Artificial Intelligence. If, due to some breakthrough, we could soon develop machines that were good translators, would we be just as dissatisfied with their intelligence as we are with the intelligence of today's store-bought chess opponents? There seem to be two classes of AI problems: problems that are unsolved and problems that, although solved, have turned out to have neither interesting nor satisfying solutions. As one task after the next becomes amenable to a machine solution, will our goal of creating machine intelligence continue to recede into the future? Once a problem is solved does it stop being an AI problem?

We think not. We are dissatisfied with the chess-playing machines not because chess-playing ability turned out to be a poorly-chosen task, but because we were unclear about what we expected. As an engineering endeavor, the creation of chess-playing machines has succeeded in that the machines play chess well. If this does not satisfy us then AI must therefore be something other than the design and development of machines that perform well on tasks generally considered to require intelligence.

Does that mean that AI is psychology? One way in which chess-playing machines fail to satisfy us is that they do not operate in a believably human way. They don't remember games. They don't analyze their failures and learn from them how to avoid similar situations in the future. Rather than say that the chess machines turned out not to be "really" intelligent, we could instead say that they do not satisfy us because they lack certain essential aspects of human intelligence. But defining Artificial Intelligence as the attempt to precisely replicate the human mind would fail to encompass a great deal of worthwhile AI research. In fact, some AI researchers, although they are interested in intelligent behavior, couldn't care less about the human mind itself. Undoubtedly there are better methods for accomplishing certain cognitive tasks than the methods used by actual humans, and those who wish to build intelligent machines should not hesitate to use them. No one other than a psychologist interested in the development of mathematical skills, for example, is likely to rush out to build a machine that computes by counting on its fingers. Yet, the human mind remains the only kind of intelligence that we can reasonably hope to study. In the human mind, we have an existence proof of a working, comprehensive intelligence.

In placing great importance on psychological realism, however, one still has to worry about computer programs, like the chess-player discussed above, that display intelligence but yet are clearly in no way related to how humans function. Are such programs **intelligent**? If one is not fully satisfied with either the engineering of intelligent-seeming programs nor with the precise modeling of the human mind as goals for AI, then one must inevitably focus on the issue of the nature of intelligence itself apart from its particular physical embodiment. One must ask "What is intelligence?" apart from whether one is talking about human, animal or machine intelligence.

Features of Intelligence

One way to examine the nature of intelligence is to list some features that we would expect an intelligent entity to have. None of these features by itself would define intelligence, indeed a being could lack any one of them and still be considered to be intelligent. Nevertheless each is an integral part of intelligence in some fundamental way. Five features of intelligence that we shall briefly discuss here are: **communication, internal knowledge, world knowledge, intentionality and creativity**.

Communication: People can communicate with an intelligent entity. We can't talk with rocks or discuss our desires with trees, no matter how hard we try. With dogs and cats we cannot express much, but we can at least express anger or pleasure. No matter how smart your dog is, he can't understand when you discuss physics with him. This does not mean that he doesn't understand anything about the physics of everyday life; it simply means he cannot communicate his knowledge. Your small child may know some physics, but discussions of that subject will have to be put in terms a child can understand. In other words, the easier it is to communicate with an entity, the more intelligent it seems. Obviously there are many exceptions. It would be unfortunate if, just because you were traveling in a country whose native language you did not speak, everyone treated you as a moron. Also, there are people who are generally considered to be intelligent but who are impossible to talk to. In general, however, communicative ability is a significant and characteristic feature of intelligence.

Internal Knowledge: We expect intelligent entities to have some knowledge about themselves. They should know what they know; they should know something about what they are thinking about. One wouldn't expect an intelligent person to partially work out a solution to a problem whose completion depends upon some piece of knowledge that he knows he doesn't possess and that he knows he cannot reasonably acquire. Probably only humans reason about their own knowledge this way; we have no way of knowing what dogs know about their own mental state. We can program computers to reason about their own knowledge in some rudimentary ways, but their behavior is not very convincing. Unfortunately, only the subjective criteria of asking questions and observing behavior seem to be useful in assessing this feature of intelligence.

World Knowledge: Intelligence also depends upon an awareness of the outside world and an ability to find and use information about the world when needed. It implies having a memory for past experiences. An intelligent entity cannot treat every experience as brand new; it must understand new experiences in terms of old ones, building up its knowledge of the world as it processes each new experience. Intelligent entities learn about the world from experience and observation.

Intentionality: Goal-driven behavior is another key feature of intelligence. Intelligent entities know what they want and, often, how to get it. An oak tree has relatively few goals, and correspondingly few plans for achieving those goals. A person living in a complex social environment has many goals and plans. Sheer number of plans, however, may not be a good measure of intelligence because a computer could easily store arbitrarily many plans. The key to intelligent intentionality has to do with how appropriately and flexibly plans can be applied to novel situations.

Creativity: We assume that every intelligent entitity has some degree of creativity. In the weakest sense, creativity might mean only finding a new route to one's food source when the old route is blocked. Creativity can also mean using a plan from one area of experience to achieve a goal in another. At its most impressive, creative behavior has changed the world by giving people a completely new framework in which to understand some class of phenomena, for example the exposition of a new scientific theory like Newtonian mechanics, or the development of a new means of artistic expression like linear perspective drawing.

If we wish to discuss artificial intelligence as opposed to general intelligence, however, we must then ask how the five features described above might be incorporated into computer programs. In other words, what problems must the field of AI address if it is to produce programs that exhibit communication, internal knowledge, world knowledge, intentionality and creativity? Although the specific problems that AI researchers have worked on have changed from time to time as some tasks have come into fashion and others have temporarily or permanently gone out of fashion, there are nevertheless a certain set of enduring problems. We now discuss ten of these problems that define the field as we see it today.

Problem 1: Representation

If people want to discuss memories, facts, objects, relationships or abstract ideas, they must have a language in which to do so. Similarly if we want a machine to be able to manipulate knowledge, the knowledge must be represented in the machine in some manipulable form. The point of representation is that computers do not manipulate ideas, facts and concepts. Conventional serial computers manipulate things like symbols, data structures and pointers. Connectionist models and other neural-like systems manipulate connection strengths and activation levels. Some future kind of computer architecture might manipulate some heretofore unenvisioned class of object. In any case, the internally-manipulated structure must be made to stand for whatever ideas, facts or concepts the system is being asked to manipulate, according to some consistent scheme that allows the processes within the machine to take on some meaning.

One kind of representation might be to store an idea as English (or other natural language) text describing the idea. Although this kind of representation makes it very easy for a person inspecting the program to figure out what is being represented, it turns out to be a very difficult representation for computer programs to manipulate. We would like problems of ambiguity, metaphor and ellipsis not to permeate our representation scheme. The representation should neither hide important similarities nor should it introduce similarities where none really exist. The English language representations: "John took an aspirin.", "John took a bus to New York." and "John took Mary to the party." are quite similar, yet the underlying concepts are not. The accidents of language usage should not confuse internal representation of concepts. On the other hand, if one concept has been stored as "John took a bus to New York" and another has been stored as "John went to New York by bus", the similarity is hidden from the system's view.

Conceptual Dependency, or CD [Schank 72] was an early attempt to represent the semantics of actions in a clear, unambiguous form. CD expressed actions in terms of a dozen conceptual primitives and their interrelationships. For example, "John took a bus to New York" would be expressed using PTRANS, the primitive for physical transfer of location:

```
(PTRANS  (actor JOHN)
         (object JOHN)
         (to  NEW-YORK)
         (instrument BUS))
```

while "John took an aspirin" would be expressed using INGEST:

```
(INGEST (actor JOHN)
        (object (MEDICINE (TYPE ASPIRIN)))
        (to (MOUTH-OF JOHN)))
```

CD is a theory of representing fairly simple actions. To express, for example, "John bet Sam fifty dollars that the Mets would win the World Series." takes about two pages of CD forms. This does not seem reasonable. We need more complex structures to handle more complex actions, and much of our work in representation has been directed at the problem of more complex events. Scripts and plans [Schank and Abelson 77, Cullingford 78, Wilensky 78] were our earliest attempts to develop larger and more comprehensive representations. Social Acts [Schank and Carbonell 78] represented the actions of institutions. Memory Organization Packets (MOPs) further extended our representations into a more hierarchical, interconnected type of memory [Schank 79, Lebowitz 80, Kolodner 80, Dyer

82] . Currently our work on representation centers on the knowledge needed to explain unexpected events. Explanation Patterns (XPs) are being developed to represent causal and explanatory knowledge [Schank 86,Kass 86,Leake and Owens 86]

Representation is a good benchmark against which to measure the progress of AI programs. "Toy" domains like blocks world [Winograd 72], [Winston 70] were feasible because they were simple enough that existing representation schemes could capture everything interesting about them. AI programs could develop without being totally blocked on representation issues. As programs have moved from these domains into more complex domains requiring real-world knowledge, however, representation has become a more difficult issue and remains one today. The brittleness and non-extensibility of many systems is due to the impoverished, sparse representation they embody.

Open topics for further work in the area of representation include the question of whether there should be one general representation scheme for objects in memory, or whether various mental processes should each use different representations for the same piece of knowledge. If the former approach is taken, then one must address the issue of how a memory representation can succinctly encode an object or event before the system knows what it wants to do with the representation, before it knows which cognitive process is going to use the representation and for which purpose.

Problem 2: Decoding

If an intelligent system has some kind of internal representation scheme, then it also must have some means of translating between what is going on in the world and what is in its internal representation. The information we get from our senses (or the information a computer or robot might get from its sensors and input devices) is not at the same level of abstraction as the conceptual processing that comprises the kind of intelligence we usually think about. Our eyes do not talk to us in terms of doorways, cliffs, onrushing cars, friends' faces or any other conceptually interesting aspects of the visual field; they tell us about patterns of light and shadow. Our ears do not tell us what words we have heard, they tell us about sounds. Taken as a whole, our senses do not directly tell us what is happening around us, yet from the information given us by our senses we can derive this information. This is what we mean by decoding.

Although we do not wish to dwell on natural language understanding issues to the exclusion of the general problem of understanding, the task of working from a natural language text to an internal representation highlights some of the problems in decoding. Consider once again the word *took* :

```
John took a bus to New York.
John took an aspirin.
John took Mary to the party.
```

Our decoding scheme must be able to interpret these very similar sentences, both using the principal verb *took*, into dissimilar internal representations.

Remembering the meaning of specific phrases like "take a car", "take a plane", "take a train", et cetera is one way this could be done, but that approach seems wasteful of memory capacity. Furthermore, it fails to capture an important generality about the use of the language. The same knowledge that allows us to understand phrases like "John took a bus." also allows us to understand specific phrases that we have probably never seen before, like "John took a hovercraft across the Channel." A more general approach to this decoding task is to have procedural knowledge associated with words. For example the procedure for decoding *took* might have, among other rules, something like:

> **''If my object is the name of some mode of transportation, then instantiate a structure representing travel. If my object is the name of a medicine, then instantiate a structure representing ingestion. If my direct object is a person and my indirect object is the name of some social occasion, then instantiate a structure representing a social date....''**

These procedures were called "requests" by [Riesbeck and Schank 76], [Birnbaum and Selfridge 79] and others. Note that the knowledge contained in a request is not just knowledge about language, it is knowledge about the world. An aspirin is not a mode of transportation. A party is a social occasion to which people customarily bring dates. This is an important point: The knowledge requirements of the decoding process cannot be isolated; the decoding process needs access to a full range of knowledge about the world. As an example of the degree to which this knowledge is removed from knowledge about language use, consider the following pair of sentences:

> **At the end of his working day, the policeman took the bus home.**
> **At the end of his working day, the bus driver took the bus back to the garage.**

One of them is about traveling someplace by bus and the other is about driving a bus someplace. It is our knowledge about policemen and bus drivers that allows us to decode these, not our knowledge about language. Our recent work on language understanding includes the MOPTRANS parser [Lytinen 84], which explores the integration of linguistic and world knowledge, and Direct Memory Access Parsing (DMAP), which makes the claim that decoding (or parsing) is an integral part of memory organization and should not be a separate process [Riesbeck and Martin 85,Martin and Riesbeck 86]

The principal problems in decoding are how memory can be arranged so that the correct knowledge is brought to bear at the correct time during the decoding process, and how the complexity of multiple, interacting, possibly conflicting requests or interpretations can be controlled.

Problem 3: Inference

Exactly where the decoding task ends and where the rest of processing begins is not clear. Continuing our saga of John on his bus, what, exactly, does the sentence "John took a bus to New York." mean? Does it mean that John went to New York? Yes. Does it mean that John is in New York now? Maybe, if we haven't heard information to the contrary. Does it mean that John wanted to go to New York? Does it mean that John is not wealthy enough to own a car? Is John a bus driver? A hijacker? There is a potentially infinite set of inferences that can be drawn from a piece of input, and a large part of intelligent understanding is the ability to draw a reasonable set of them.

One way to make inferences is via inference rules and pattern matching. A system can have a library of rules, each of which has a precondition side and an inference side. When the precondition side is matched, the inference is generated. An example of a such a rule could be: "If someone travels to someplace, infer that they had some business there" or "If someone travels by bus, assume they do not own a car." These rules can be chained together by matching the inference of one rule against the preconditions of the next, so that a system could answer "Is John Poor?" by chaining from "Poor people do not own cars" to "People who do not own cars travel by bus" to "John went to New York by bus". This inference may or may not be correct, there are also many other inferences that could potentially have been made at the same time from the same input. How well AI programs make reasonable inferences, how well they decide which inferences to make and how they decide how far to extend an inference chain are good ways to measure our progress against the inference problem.

The inferencing task highlights the importance of a good representation scheme. The representation for a class of events, for example travel, is a good place to index, in memory, the inferences we wish to make whenever we encounter an event of that class. This approach was taken by [Rieger 75]. If our representation groups together the events about which we wish to make similar inferences, then it is serving the inference task well. If it scatters similar events throughout memory then it is not.

Problem 4: Control of Combinatorial Explosion

The problem with inferences is that there are too many of them to make. If, for example, we can make five inferences from a fact, and five more inferences from each of those inferences and so forth, then the combinatorial complexity of trying to investigate inference chains of any length greater than a few steps becomes overwhelming. Processing power is not infinite, either in people or in machines.

Conceptual Dependency avoided the worst of this problem by representing the world at a coarse enough grain size that the number of inferences remained small. The problem was nevertheless there. A more general solution to this problem goes hand in hand with a solution to the problem of representations being too low-level: As we represent more complex events and objects, we need to constrain the undirected inferencing that we ask our systems to do. We need to structure knowledge at a higher level of abstraction.

The scripts used in SAM and FRUMP [Cullingford 78,DeJong 79] were an attempt at this kind of structuring. The point of a script was to package together a stereotyped sequence of events along with some inferences so that the whole entity could be treated as one representation object. An example is the restaurant script, which had in it CD forms corresponding to the sequence of actions one typically expected at a restaurant, such as entering, being seated, ordering, eating, paying, leaving a tip and leaving. While sitting in a restaurant, when the waiter arrives at our table, all we need to do is recognize a script element rather than try to figure out who this man is and why he is there. We know that we can order food or ask questions about the menu because the restaurant script already contains, frozen and ready to use, the inferences we will need. We don't need to search memory for inference rules in order to understand the sequence of events. For routine stories, scripts reduced the general problem of understanding to a problem of matching a script element. A more general overview of the basic idea of this kind of knowledge structuring and packaging can be found in [Minsky 75]. More recently, more hierarchical, modular and abstract knowledge structures have been developed in response to some of the shortcomings of scripts [Schank 82].

As an example of the kind of control an AI program must have over its inferencing even in relatively simple stories, consider the following example, suggested by Larry Birnbaum:

> An undercover agent was tailing a suspect. The suspect walked into a hardware store. The agent followed. After a few minutes in the store, the suspect

began looking nervously at the agent, who was trying to look inconspicuous next to a display of ropes. The agent measured off some rope, took it to the counter, paid for it and left the store.

If we ask our understanding system why the agent bought the rope, we want it to realize that he didn't want the rope at all. He merely bought the rope because that is a typical, inconspicuous thing to do in a hardware store and because he happened to be standing next to the rope when he became aware that his quarry was nervous about him. No reasonable amount of inference off the low-level features of the scene could yield this understanding; a system would never get to a state of understanding by starting from hardware stores or rope. At the same time, nothing in the Undercover Agent knowledge structures is going to say anything specific about buying rope. Reasoning from the high-level construct of "looking inconspicuous" to match against the action of buying the rope places demands upon the tasks of matching and instantiation that we don't know how to solve without asking the understanding system to search exhaustively through everything that it knows.

Problem 5: Indexing

The difference between reasoning from primitive inference rules and reasoning from cases or larger knowledge structures like scripts can be summarized as the difference between building a causal model and retrieving one. Presumably, saving one's analytic work and retrieving it later requires less processing than trying to develop a model from scratch. Once we know that we are in the restaurant script, for example, we can easily place in context the appearance at our table of a man with a notepad.

If there is to be any advantage to retrieving complex causal models and other knowledge structures rather than building them from scratch, then there must be an efficient way to index structures so that the right one can be retrieved at the right time. If, on the other hand, it requires just as much work to look through memory to find a knowledge structure as it would to derive it from scratch, then memory must be reorganized so that the offending knowledge structure can be found more quickly.

A way to look at this memory indexing process in people is to examine the phenomenon of reminding. A favorite reminding story is the "Steak and the Haircut" reminding, reported in [Schank 82]. Two friends were talking and one mentioned that he could never get his steak as rare as he wanted it. The other replied that that reminded him of a period of time, years ago, when he could never get his hair cut as short as he wanted. The two episodes have little in common at the surface level, yet they share the thematic similarity of someone being unable to get something as extreme as he wanted it. In order to get reminded of the second story by the first, the listener must first have abstracted from the details of the steak story to the underlying pattern (people not believing you when you ask for something extreme?), and then retrieved the haircut story, which was indexed under that abstraction.

CYRUS [Kolodner 80] focused on the issue of how memory could be organized and indexed in order to answer questions whose answers were indirectly available within memory but which were not directly answered by any one fact in memory.

Some aspects of the knowledge structure indexing and selection problem can be solved easily and some cannot. With scripts, for example, physical location and stereotyped objects play a strong role. If we hear about someone going into a restaurant, we can begin to infer that they will order a meal. If someone picks up a telephone, we can assume they will make a call. Of course reasoning from physical location is only a rough guideline, and it is often incorrect. A system embodying this type of script selection would be tripped up by the following story:

> **"John needed to make a phone call. He had no change in his pocket. Down the block was a diner. He walked into it."**

A system reasoning from physical locations would assume that John was about to order a meal. Furthermore, the more abstract patterns, which are the ones that buy a great deal of processing and inferential power, are the most difficult to index so that features derived from a story would be good retrieval cues. How would a system notice an instance of revenge? Of a double-cross? Of a love triangle? Indexing abstract patterns so that they can be retrieved when appropriate is an important memory organization problem to solve.

Problem 6: Prediction and Recovery

A good example of how these structures and patterns are applied to the task of understanding is found in the processes of expectation and prediction. If an understanding system is constantly trying to use its active knowledge structures to predict what will occur next, then the system is in effect no longer asking "what is going on here?" It is instead asking "How does this event match the expectations that have been set up by my active knowledge structures." This is a narrower, more specific question, and it is easier to see how a system might answer it. Given a good set of predictions, a predictive understander's job is to match input against its predictions rather than to analyze the input from scratch.

Of course these predictions are not always going to be correct. Sometimes they will be wrong because the understander has misunderstood what is actually happening and has activated the wrong knowledge structure. Sometimes they will be wrong because, although the right knowledge structure is active, a genuinely bizarre event has occurred. Or the prediction could have been fulfilled, but in some way that the understander failed to notice. When people fail in an attempt to understand something, they do not simply give up and spout an error message, they attempt to recover from the failure and continue processing. Obviously, our machines should do the same.

Predictive failures are not the only kinds of failures from which people can recover. Plan execution failure occurs when a plan step either fails to achieve its desired goal, is unexecutable, or has undesirable and unexpected side effects. Memory retrieval failure occurs when we look for a knowledge structure to package a certain event or set of events and are unable to find one.

Recovering from any of these types of failures by backtracking is a non-theory. For one reason, the combinatorics of this approach are impossible. Furthermore, the approach is not executable in real time under real resource constraints. Once a robot has walked off a cliff, it cannot backtrack and try its next option. A system must anticipate and avoid plan failures, and it must take advantage of the learning opportunity offered by failure. Failure conveys a tremendous amount of information in that properly analyzing a failure suggests not only appropriate recovery strategies, but also strategies for avoiding the failure-inducing situation in the future. Failure allows us to learn new predictive knowledge structures, new planning rules or new memory retrieval strategies. The CHEF program [Hammond 86] explored this process in the planning domain.

At the simplest level of failure recovery, a knowledge structure can have rules that specify, if a certain type of failure occurs, to use a different knowledge structure. For xample, if we walk into a restaurant and the expected hostess does not materialize but there is a counter, we can assume that we are in a fast-food restaurant and activate the fast-food restaurant knowledge structure with its different set of expectations, inferences and event sequences. As we shall explain below, a more sophisticated failure recovery strategy is available as well, one that has the added benefit of enabling a system to learn new knowledge structures.

Problem 7: Dynamic Modification

Of course none of the knowledge and memory organization needed for understanding is built in; it must be acquired. Humans change as a result of their experience, augmenting their store of knowledge, reorganizing their memories, forming generalizations. Learning is a quintessentially human capability, yet, aside from an occasional peripheral appearance it has been conspicuously absent from AI programs until very recently. A program was limited to the stock of knowledge with which it started, which meant that our understanding systems could be given the same story over and over again, and the program would treat each occurrence as though it were reading a totally new story, gaining neither any efficiency in understanding nor any generalized knowledge as a result of its experience.

Earlier we showed how getting reminded of a relevant knowledge structure was a lazy way to understand a new episode in that, to the extent that the new episode was similar to the class of old episodes represented by the knowledge structure, a system could understand by "filling in the blanks" in the knowledge structure rather than by doing new analysis from scratch. Similarly, a lazy way to learn new knowledge structures is to modify old ones slightly in order to fit new data.

A good model for this kind of knowledge structure revision is explanation. When people have a predictive failure in understanding, they are surprised. Things aren't as they expected. They recover by explaining to themselves what is going on, which results in both an understanding of the current situation and a revised set of predictive knowledge structures to use in the future.

At Yale, we have been studying this phenomenon by asking people to make up explanations in response to the Swale story. Swale was a well-known and extremely successful three-year-old race horse who was suddenly and unexpectedly found dead in his stall. Although no explanation was immediately offered in the newspapers, certain possible explanations were frequently given. Maybe Swale was killed for the insurance money. (This turned out to be improbable, as he was under-insured.) Maybe he was killed by the owner of a competitor. (No evidence of foul play was discovered.) Maybe, like Jim Fixx, the author of books on jogging who died of a heart attack, he had some kind of hidden heart defect that was brought out by the strain of competing in horse races.

The point of mentioning these explanations is that they were not developed from scratch by chaining together primitive rules. People seem to explain by making reference to remindings, or to frozen patterns of causal reasoning. We have been calling the frozen causal patterns Explanation Patterns, or XPs. Our approach is that the key to explanation is getting reminded of a relevant XP and applying it rather than trying to build an explanation from scratch. This approach controls what would otherwise be an inference problem with impossible combinatorics.

Making an explanation allows an understander to focus on which features of an episode are causally relevant and therefore interesting. Implicit in our knowledge structures and knowledge packaging relationships is the notion that causal relationships underlie the packaging links. If causality is the glue that holds memory packages together, then causal analysis of input episodes is essential to the task of analyzing a predictive failure and using it to modify the predictive knowledge structures responsible for it.

Trying to get a system to make explanations opens up a set of interesting problems. Where should the knowledge derived from the explanation be indexed, so that the predictive or other failure could be prevented? How far should the explanation be generalized? How should explanation patterns be indexed in memory? How should input episodes be analyzed to select features that would be likely retrieval cues.

Problem 8: Generalization

Another kind of learning comes not from analyzing failures, but from noticing regularities and patterns in the world and drawing generalizations from them. If people see the same sets of features co-occurring often enough, they will learn the generalization that they occur together. Winston's concept-learning program [Winston 70] used positive and negative examples to induce the definitions of structures in the blocks-world domain. The IPP program [Lebowitz 80] augmented its store of predictive knowledge structures by noticing and remembering the generalities across stories that it read.

Noticing regularities is a good source of anomalies that, once explained, can be added to a system's store of knowledge. A problem with using induction without explanation is that it is not particularly powerful compared to the kind of concept learning that people actually do. On the one hand people notice generalities in one or two trials that an inductive machine would take much longer to notice. On the other hand inductive machines, simply by correlating features, make generalizations that people would never make. IPP, for example, could erroneously "learn" that bombings in India always kill exactly two people, simply by saving the features that multiple stories happened to have in common. Whenever possible, forming generalizations by using explanations to identify causally significant features is a stronger method.

Problem 9: Curiosity

As a combined experiment and exercise we once presented to a class of graduate students, a newspaper story about a teenage suicide car bomber who had been captured before he had the chance to destroy his target. We presented the story one sentence at a time, asking the students at each pause to propose questions that they would like to see answered. The questions were exactly as would be suggested by a model of understanding based upon prediction and substantiation of predictions. At each pause, listeners were ready with questions directed towards confirming, modifying or refining a partial theory of what the story was about. Although graduate students in a scientific discipline are probably more explicit about forming and evaluating hypotheseses than are most people, their behavior is still representative some general process.

Cats, small children, and even some adults, are curious. They ask questions about what they see, they wonder about what they hear, and they object to what they are told. This is a natural result of a system making predictions and wondering why those predictions fail. One view of curiosity is that a system is curious if it devotes processing effort to identifying gaps in its knowledge base and seeks to fill them.

Our work on explanation [Schank 86] deals with some questions pertaining to curiosity, namely how a system should identify what is worth wondering about and how hypotheses should be generated and tested. We have also explored the use of curiosity, the underlying stimulus for learning, in education itself [Schank and Slade 85]. .

Problem 10: Creativity

One story that was facetiously offered in explanation of the abovementioned Swale episode was that Swale died from a heart attack resulting from the excitement of anticipating what his future life on the stud farm would be like. Although this is a ridiculous explanation from a reasonable, factual point of view, we nevertheless like it because it somehow seems funny and creative.

What we see in these creative explanations is the ability to **intelligently misapply** explanation patterns. A creative explainer can start with an inappropriate pattern, for example the folklore that too much sex can result in early death. Although on the face of it, this pattern cannot apply to race horses because they are typically not bred until they are retired from racing, it is a departure point for modification. It is reasonable that indexing into memory with race horses and sex should retrieve knowledge about stud farms, which, although not immediately applicable, can be relevant. Since the stud farm phase of a horse's life occurs after his active racing career, and since Swale died at the peak of his career, there cannot be a direct causal connection. Future events can, however influence present events via the effects of anticipation. This is a general rule that is useful, for example, in analyzing stock price movements in reaction to events that are predicted but have not yet happened. Reasoning about the effects of anticipating the stud farm, an explainer could predict excitement. Excitement can be connected to death via stress-induced heart attacks.

By taking an inappropriate Explanation Pattern and modifying it to fit the current episode, we derive an explanation that is creative and humorous. Of course the same process can yield useful explanations as well. The use of the Jim Fixx reminding to explain Swale's death is also a creative act. Using an explanation about joggers to explain race horse death is an example of the same kind of misapplication and modification that we discussed above. The explanation cannot apply at first because Swale, not being a human, cannot be a jogger. A creative explanation system could, however, use the causal knowledge about how being a jogger connected to Jim Fixx's death. This knowledge lets the program ask the question: "Is there anything a race horse does that has the same causal significance to Swale as jogging held to Jim Fixx's death, namely that it was a source of physical exertion?" With this narrow a search task, it is not difficult to derive the explanation of a heart attack brought on by excessive physical strain involved in racing. This adaptation of inappropriate patterns is at the heart of what we wish to study in creativity. This work is reported in more depth in [Schank 86].

Creativity, whether in the artistic, scientific or problem-solving domain, involves using something in a novel and unexpected way. A creative piece of engineering design might involve using a part, structure or material in a new way. Similarly, creative problem solving might involve using a technique from one field in solution of a problem in another. Creative explanation and understanding involves taking a pattern or knowledge structure and modifying it to fit events other than the ones for which it was originally derived.

The problem of creative reasoning, which at first seems mystical and far removed from that which can be implemented on a machine, reduces to two sub-problems, retrieval and adaptation. These are difficult problems, but they are in principle amenable to a mechanistic approach. Although it is tempting to say that being creative means relaxing the constraints on retrieving and matching memories, it is not this simple. Human and machine processing power is not infinite so randomly trying one inappropriate pattern after the next is schizophrenic rather than creative. The important aspect of creative understanding is to be intelligent about which "inappropriate" structures to use: to have a memory that returns a reasonable set of near-miss candidate patterns.

Conclusions

Which problems are most important? All of them are important of course. But the tasks associated with learning must pervade and inform work on the other problems. AI has made considerable progress in defining what it is that is learned. Now it is time to demand actual learning of our programs.

The problems we have listed here are not meant as the ultimate definition of what Artificial Intelligence is or should be. They are instead a set of goals and benchmarks that we might use. We do not want to argue that people in the field should work directly on these more abstract issues rather than on programs that addresses a particular concrete task. Setting out to write a program that performs well on some real task, for example, translating languages or learning how to play chess, is essential. It is probably a more reasonable research strategy than trying directly and abstractly to write a program that "uses remindings" or "acts curious". But, letting work towards concrete tasks be guided and informed by these abstract problems, and measuring completed AI work in terms of how well it addresses them, is a way to avoid the problem of ending up with programs that performs well without having anything at all interesting to say about intelligence or mind.

Acknowledgements

Thanks to Stephen Slade and David Leake for helpful comments on drafts of this manuscript.

References

[Birnbaum and Selfridge 79] Birnbaum, L., aud Selfridge, M., ***Problems in Conceptual Analysis of Natural Language,*** Technical Report 168, Yale University Department of Computer Science, October 1979.

[Cuilingford 78] Cullingford, R., *Script Application: Computer Understanding of Newspaper Stories,* Ph.D. Thesis, Yale University, 1978. Technical Report 116.

[DeJong 79] DeJong, G., *Skimming Stories in Real Time: An Experiment in IntegratedUnderstanding,* Ph.D. Thesis, Yale University, May 1979. Technical Report 158.

[Dyer 82] Dyer, M., *IN-DEPTH UNDERSTANDING: A Computer Model of Integrated Processing For Narrative Comprehension,* Technical Report 219, Yale University Department of Computer Science, May 1982.

[Hammond 86] Hammond, K.J., *Case-based Planning: An Integrated Theory of Planning, Learning and Memory,* Ph.D. Thesis, Yale University, 1986.

[Kass 86] Kass, A., Modifying Explanations to Understand Stories, *Proceedings of the Eighth Annual Conference of the Cognitive Sci-*

ence Society, Cognitive Science Society, Amherst, MA, August 1986.

[Kolodner 80] Kolodner, J.L., *Retrieval and Organizational Strategies in Conceptual Memory: A Computer Model,* Ph.D. Thesis, Yale University, November 1980. Technical Report 187.

[Leake and Owens 86] Leake, D., and Owens, C., Organizing Memory for Explanation, *Proceedings of the Eighth Annual Conference of the Cognitive Science Society,* Cognitive Science Society, Amherst, MA, August 1986.

[Lebowitz 80] Lebowitz, M., *Generalization and Memory in an Integrated Understanding System,* Ph.D. Thesis, Yale University, October 1980. Technical Report 186.

[Lytinen 84] Lytinen, S., *The Organization of Knowledge In a Multi-lingual, Integrated Parser,* Ph.D. Thesis, Yale University, 1984. Technical Report 340.

[Martin and Riesbeck 86] Martin, Charles E. and Riesbeck, Christopher K., Uniform Parsing and Inferencing for Learning, *Proceedings of the Fifth National Conference on Artificial Intelligence,* AAAI, Philadelphia, PA, August 1986, pp. 257-261.

[Minsky 75] Minsky, M., A framework for representing knowledge, P. Winston ed., *The Psychology of Computer Vision,* McGraw-Hill, New York, 1975, chapter 6, pages 211-277.

[Rieger 75] Rieger, C., Conceptual Memory and Inference, *Conceptual Information Processing,* North-Holland, Amsterdam, 1975.

[Riesbeck and Martin 85] Riesbeck, C.K., and Martin, C.E., *Direct Memory Access Parsing,* Technical Report 354, Yale University Department of Computer Science, January 1985.

[Riesbeck and Schank 76] Riesbeck, C., and Schank, R.C., *Comprehension by Computer: Expectation-based Analysis of Sentences in Context,* Technical Report 78, Yale University Department of Computer Science, October1976.

[Schank and Abelson 77] Schank, R.C. and Abelson, R., *Scripts, Plans, Goals and Understanding,* Lawrence Erlbaum Associates, Hillsdale, New Jersey, 1977.

[Schank and Carbonell 78] Schank, R.C., and Carbonell, J., *Re: The Gettysburg Address. Representing Social and Political Acts,* Technical Report 127, Yale University, 1978.

[Schank and Slade 85] Schank, R.C., and Slade, S., *Education and Computers: An AI Perspective,* Technical Report 431, Yale University Department of Computer Science, October 1985.

[Schank 72] Schank, R.C., *Conceptual Dependency: A Theory of Natural Language Understanding,* Cognitive Psychology, 3/4 (1972), pp. 552-631.

[Schank 79] Schank, R.C., *Reminding and Memory Organization: An Introduction to MOPs,* Technical Report 170, Yale University Department of Computer Science, 1979.

[Schank 82] Schank, R.C., *Dynamic Memory: A Theory of Learning in Computers and People,* Cambridge University Press, 1982.

[Schank 86] Schank, R.C., *Explanation Patterns: Understanding Mechanically and Creatively,* Lawrence Erlbaum Associates, Hillsdale, NJ, 1986.

[Wilensky 78] Wilensky, R., *Understanding Goal-Based Stories,* Ph.D. Thesis, Yale University, 1978. Technical Report 140.

[Winograd 72] Winograd, T., *Understanding Natural Language,* Academic Press, New York, 1972.

[Winston 70] Winston, P.H., *Learning Structural Descriptions from Examples,* Technical Report 231, AI Laboratory, Massachusetts Institute of Technology, 1970. Reprinted in P.Winston, Ed., (*The Psychology of Computer Vision*), 1975, McGraw-Hill.

END
9-87
DTIC